

Part 2:

Code Provided:

```
package tsp;

import java.util.*;

public class tsp implements Runnable
{
    int size;
    int routes[][];//edges
    String cities[];//vertices

    Thread runners[];
    int threadCompleteCount;

    String solution;
    int totDistance;

    tsp()
    {
        try
        {
            int i, j;
            String choice;

            Scanner scan = new Scanner(System.in);
            System.out.println("Enter the number of cities ");

            size = scan.nextInt();
            scan.skip("\n");

            cities = new String[size];
            routes = new int[size][size];

            System.out.println("Set city names : ");
            for(i=0; i< size; i++)
            {
                System.out.println("City " + (i+1) );
                cities[i] = scan.nextLine();
            }

            System.out.println("Set interconnecting routes ");
            for(i =0; i< size; i++)
            {

                for(j =i+1; j<size; j++ )
                {
                    System.out.println("Is there a route between " + cities[i] + " and " + cities[j] + "(y/n) : " );
                    choice = scan.nextLine();
                    if(choice.equalsIgnoreCase("y"))
                    {

```

```

        System.out.println("Enter distance : ");
        routes[i][j] = routes[j][i] = scan.nextInt();
        scan.skip("\n");
    }
    else
    { //no route
        routes[i][j] = routes[j][i] = 999;
    }
}
//for(j
}//for(i

threadCompleteCount = 0;
solution = "Nearest Neighbour Algorithm couldnt form a tour to visit all
cities";
totDistance = 999;

runners = new Thread[size];
for(i =0 ; i< size; i++)
{
    runners[i]= new Thread(this, String.valueOf(i));
    runners[i].start();
}
}
catch(Exception ex)
{
    System.out.println("Err : "+ ex);
}
}//tsp()

void display()
{
    int i, j;
    for(i = 0; i< size; i++)
    {
        System.out.println();
        System.out.print(cities[i] + " : ");
        for(j =0; j< size; j++)
        {
            System.out.print( cities[j] + "(" + routes[i][j] + ")   ");
        }
    }
    System.out.println();
}

public void run()
{
    int sPos = Integer.parseInt(Thread.currentThread().getName());
    solvetspUsingNearestNeighbour(sPos);
    threadCompleteCount++;
}

boolean solvetspUsingNearestNeighbour(int startPos)
{

```

```

boolean isTourComplete = false;
try
{
    String solution = "";
    int i, j, min, currentPos, nextPos, totDistance;
    int visitedCities[];
    int vi;

    //initializations and allocations

    visitedCities = new int[size];
    vi = 0 ;
    totDistance = 0;

    //mark startPos as visited
    visitedCities[vi] = startPos;
    vi++;

    solution = cities[startPos];

    currentPos = startPos;
    //tour

    while(! isTourComplete)
    {
        nextPos = -1;
        min = 999;

        for(i = 0; i < size; i++)
        {
            if(routes[currentPos][i] != 999 && currentPos != i)
            {
                int flag = 0;

                //check for being unvisited
                for(j = 0; j < vi; j++)
                {
                    if(visitedCities[j] == i)
                    {
                        flag = 1;
                        break;
                    }
                }
                //for(j ...

                if(flag == 0)
                {//unvisited
                    if(routes[currentPos][i] < min)
                    {
                        min = routes[currentPos][i];
                        nextPos = i;
                    }
                }
                //if(routes...
            }
        }
        //for(i ...
    }
}

```

```

        if(nextPos != -1)
        {//move to next city
            totDistance += min;
            visitedCities[vi] = nextPos;
            vi++;
            solution = solution + " " + cities[nextPos];
            currentPos = nextPos;
        }
        else
        {
            break;
        }

        if(vi == size)
        {
            //tour back to start city
            if(routes[currentPos][startPos] != 999)
            {
                solution = solution + " " + cities[startPos];
                totDistance += routes[currentPos][startPos];
                isTourComplete = true;
                if(totDistance < this.totDistance)
                {
                    this.totDistance = totDistance;
                    this.solution = solution + "\nTotal Distance : " +
totDistance;
                }
            }
            else
            {
                isTourComplete = false;
                break;
            }
        }
    }//while

}
catch(Exception ex)
{
    solution = "Err : " + ex.getMessage();
}
return isTourComplete;
}

void displaySolution()
{
    while(threadCompleteCount < size)
    {
        try
        {
            Thread.sleep(1000);
        }
        catch(Exception ex)
    }
}

```

```

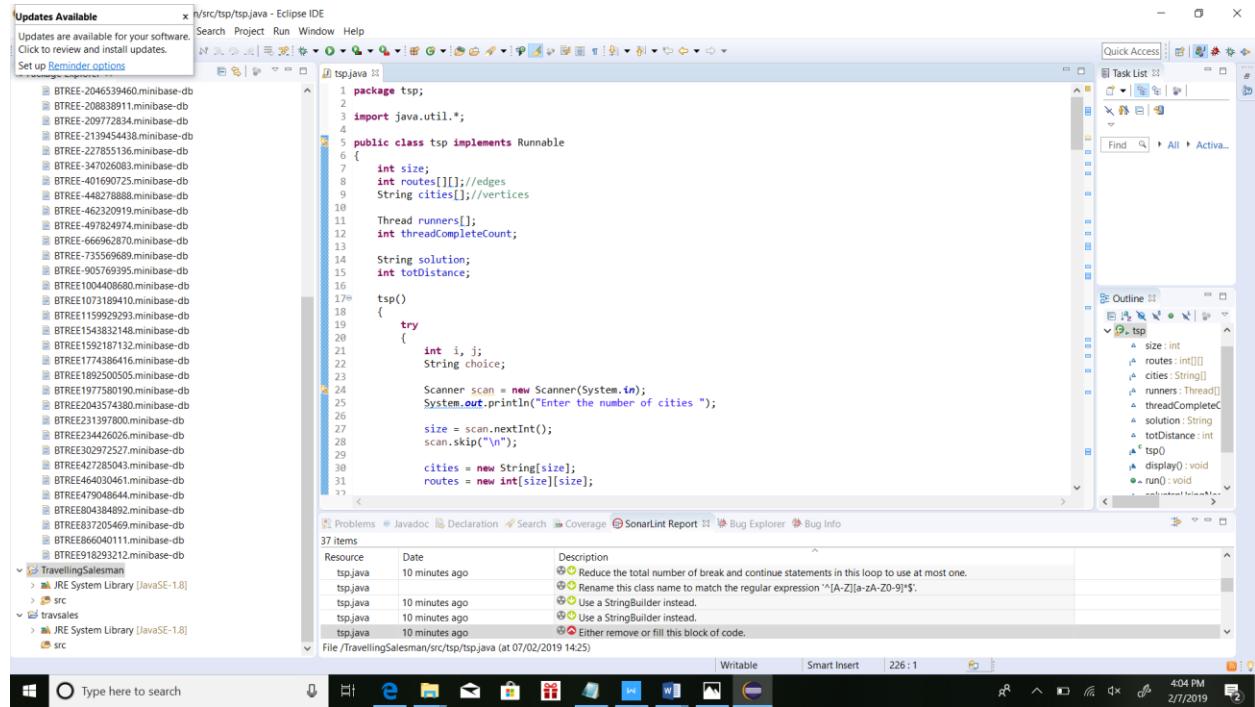
        {
    } //while
    System.out.println("Solution : " + solution);
}

public static void main(String[] args)
{
    tsp tsp = new tsp();
    tsp.display();
    tsp.displaySolution();

}

```

Screenshots of Code:



Updates Available

Updates are available for your software.
Click to review and install updates.

Set up [Reminder options](#)

```

27     size = scan.nextInt();
28     scan.skip("\n");
29
30     cities = new String[size];
31     routes = new int[size][size];
32
33     System.out.println("Set city names : ");
34     for(i=0; i < size; i++)
35     {
36         System.out.print("City " + (i+1) );
37         cities[i] = scan.nextLine();
38     }
39
40     System.out.println("Set interconnecting routes ");
41     for(i =0; i < size; i++)
42     {
43
44         for(j =i+1; j < size; j++)
45         {
46             System.out.println("Is there a route between " + cities[i] + " and " + cities[j] + "(y/n) : " );
47             choice = scan.nextLine();
48             if(choice.equalsIgnoreCase("y"))
49             {
50                 System.out.println("Enter distance : ");
51                 routes[i][j] = routes[j][i] = scan.nextInt();
52                 scan.skip("\n");
53             }
54             else
55             { //no route
56                 routes[i][j] = routes[j][i] = 999;
57             }
58         }
59     }
60
61     threadCompleteCount = 0;
62     solution = "Nearest Neighbour Algorithm couldnt form a tour to visit all cities";
63     totDistance = 999;
64
65     runners = new Thread[size];
66     for(i =0; i < size; i++)
67     {
68         runners[i] = new Thread(this, String.valueOf(i));
69         runners[i].start();
70     }
71
72     } catch(Exception ex)
73     {
74         System.out.println("Err : "+ ex);
75     }
76 } //tsp()
77
78 void display()
79 {
80     int i, j;
81     for(i = 0; i < size; i++)
82     {
83         System.out.println();
84         System.out.print(cities[i] + " : ");
85         for(j =0; j < size; j++)
86         {
87             System.out.print(routes[i][j] + " ");
88         }
89     }
90 }
91
92 void run()
93 {
94     int i, j;
95     for(i = 0; i < size; i++)
96     {
97         for(j =0; j < size; j++)
98         {
99             if(routes[i][j] == 999)
100            routes[i][j] = routes[j][i] = 999;
101        }
102    }
103 }
104
105 void run0()
106 {
107     int i, j;
108     for(i = 0; i < size; i++)
109     {
110         for(j =0; j < size; j++)
111         {
112             if(routes[i][j] == 999)
113            routes[i][j] = routes[j][i] = 999;
114        }
115    }
116 }
117
118 void run1()
119 {
120     int i, j;
121     for(i = 0; i < size; i++)
122     {
123         for(j =0; j < size; j++)
124         {
125             if(routes[i][j] == 999)
126            routes[i][j] = routes[j][i] = 999;
127        }
128    }
129 }
130
131 void run2()
132 {
133     int i, j;
134     for(i = 0; i < size; i++)
135     {
136         for(j =0; j < size; j++)
137         {
138             if(routes[i][j] == 999)
139            routes[i][j] = routes[j][i] = 999;
140        }
141    }
142 }
143
144 void run3()
145 {
146     int i, j;
147     for(i = 0; i < size; i++)
148     {
149         for(j =0; j < size; j++)
150         {
151             if(routes[i][j] == 999)
152            routes[i][j] = routes[j][i] = 999;
153        }
154    }
155 }
156
157 void run4()
158 {
159     int i, j;
160     for(i = 0; i < size; i++)
161     {
162         for(j =0; j < size; j++)
163         {
164             if(routes[i][j] == 999)
165            routes[i][j] = routes[j][i] = 999;
166        }
167    }
168 }
169
170 void run5()
171 {
172     int i, j;
173     for(i = 0; i < size; i++)
174     {
175         for(j =0; j < size; j++)
176         {
177             if(routes[i][j] == 999)
178            routes[i][j] = routes[j][i] = 999;
179        }
180    }
181 }
182
183 void run6()
184 {
185     int i, j;
186     for(i = 0; i < size; i++)
187     {
188         for(j =0; j < size; j++)
189         {
190             if(routes[i][j] == 999)
191            routes[i][j] = routes[j][i] = 999;
192        }
193    }
194 }
195
196 void run7()
197 {
198     int i, j;
199     for(i = 0; i < size; i++)
200     {
201         for(j =0; j < size; j++)
202         {
203             if(routes[i][j] == 999)
204            routes[i][j] = routes[j][i] = 999;
205        }
206    }
207 }
208
209 void run8()
210 {
211     int i, j;
212     for(i = 0; i < size; i++)
213     {
214         for(j =0; j < size; j++)
215         {
216             if(routes[i][j] == 999)
217            routes[i][j] = routes[j][i] = 999;
218        }
219    }
220 }
221
222 void run9()
223 {
224     int i, j;
225     for(i = 0; i < size; i++)
226     {
227         for(j =0; j < size; j++)
228         {
229             if(routes[i][j] == 999)
230            routes[i][j] = routes[j][i] = 999;
231        }
232    }
233 }
234
235 void run10()
236 {
237     int i, j;
238     for(i = 0; i < size; i++)
239     {
240         for(j =0; j < size; j++)
241         {
242             if(routes[i][j] == 999)
243            routes[i][j] = routes[j][i] = 999;
244        }
245    }
246 }
247
248 void run11()
249 {
250     int i, j;
251     for(i = 0; i < size; i++)
252     {
253         for(j =0; j < size; j++)
254         {
255             if(routes[i][j] == 999)
256            routes[i][j] = routes[j][i] = 999;
257        }
258    }
259 }
260
261 void run12()
262 {
263     int i, j;
264     for(i = 0; i < size; i++)
265     {
266         for(j =0; j < size; j++)
267         {
268             if(routes[i][j] == 999)
269            routes[i][j] = routes[j][i] = 999;
270        }
271    }
272 }
273
274 void run13()
275 {
276     int i, j;
277     for(i = 0; i < size; i++)
278     {
279         for(j =0; j < size; j++)
280         {
281             if(routes[i][j] == 999)
282            routes[i][j] = routes[j][i] = 999;
283        }
284    }
285 }
286
287 void run14()
288 {
289     int i, j;
290     for(i = 0; i < size; i++)
291     {
292         for(j =0; j < size; j++)
293         {
294             if(routes[i][j] == 999)
295            routes[i][j] = routes[j][i] = 999;
296        }
297    }
298 }
299
300 void run15()
301 {
302     int i, j;
303     for(i = 0; i < size; i++)
304     {
305         for(j =0; j < size; j++)
306         {
307             if(routes[i][j] == 999)
308            routes[i][j] = routes[j][i] = 999;
309        }
310    }
311 }
312
313 void run16()
314 {
315     int i, j;
316     for(i = 0; i < size; i++)
317     {
318         for(j =0; j < size; j++)
319         {
320             if(routes[i][j] == 999)
321            routes[i][j] = routes[j][i] = 999;
322        }
323    }
324 }
325
326 void run17()
327 {
328     int i, j;
329     for(i = 0; i < size; i++)
330     {
331         for(j =0; j < size; j++)
332         {
333             if(routes[i][j] == 999)
334            routes[i][j] = routes[j][i] = 999;
335        }
336    }
337 }
338
339 void run18()
340 {
341     int i, j;
342     for(i = 0; i < size; i++)
343     {
344         for(j =0; j < size; j++)
345         {
346             if(routes[i][j] == 999)
347            routes[i][j] = routes[j][i] = 999;
348        }
349    }
350 }
351
352 void run19()
353 {
354     int i, j;
355     for(i = 0; i < size; i++)
356     {
357         for(j =0; j < size; j++)
358         {
359             if(routes[i][j] == 999)
360            routes[i][j] = routes[j][i] = 999;
361        }
362    }
363 }
364
365 void run20()
366 {
367     int i, j;
368     for(i = 0; i < size; i++)
369     {
370         for(j =0; j < size; j++)
371         {
372             if(routes[i][j] == 999)
373            routes[i][j] = routes[j][i] = 999;
374        }
375    }
376 }
377
378 void run21()
379 {
380     int i, j;
381     for(i = 0; i < size; i++)
382     {
383         for(j =0; j < size; j++)
384         {
385             if(routes[i][j] == 999)
386            routes[i][j] = routes[j][i] = 999;
387        }
388    }
389 }
390
391 void run22()
392 {
393     int i, j;
394     for(i = 0; i < size; i++)
395     {
396         for(j =0; j < size; j++)
397         {
398             if(routes[i][j] == 999)
399            routes[i][j] = routes[j][i] = 999;
400        }
401    }
402 }
403
404 void run23()
405 {
406     int i, j;
407     for(i = 0; i < size; i++)
408     {
409         for(j =0; j < size; j++)
410         {
411             if(routes[i][j] == 999)
412            routes[i][j] = routes[j][i] = 999;
413        }
414    }
415 }
416
417 void run24()
418 {
419     int i, j;
420     for(i = 0; i < size; i++)
421     {
422         for(j =0; j < size; j++)
423         {
424             if(routes[i][j] == 999)
425            routes[i][j] = routes[j][i] = 999;
426        }
427    }
428 }
429
430 void run25()
431 {
432     int i, j;
433     for(i = 0; i < size; i++)
434     {
435         for(j =0; j < size; j++)
436         {
437             if(routes[i][j] == 999)
438            routes[i][j] = routes[j][i] = 999;
439        }
440    }
441 }
442
443 void run26()
444 {
445     int i, j;
446     for(i = 0; i < size; i++)
447     {
448         for(j =0; j < size; j++)
449         {
450             if(routes[i][j] == 999)
451            routes[i][j] = routes[j][i] = 999;
452        }
453    }
454 }
455
456 void run27()
457 {
458     int i, j;
459     for(i = 0; i < size; i++)
460     {
461         for(j =0; j < size; j++)
462         {
463             if(routes[i][j] == 999)
464            routes[i][j] = routes[j][i] = 999;
465        }
466    }
467 }
468
469 void run28()
470 {
471     int i, j;
472     for(i = 0; i < size; i++)
473     {
474         for(j =0; j < size; j++)
475         {
476             if(routes[i][j] == 999)
477            routes[i][j] = routes[j][i] = 999;
478        }
479    }
480 }
481
482 void run29()
483 {
484     int i, j;
485     for(i = 0; i < size; i++)
486     {
487         for(j =0; j < size; j++)
488         {
489             if(routes[i][j] == 999)
490            routes[i][j] = routes[j][i] = 999;
491        }
492    }
493 }
494
495 void run30()
496 {
497     int i, j;
498     for(i = 0; i < size; i++)
499     {
500         for(j =0; j < size; j++)
501         {
502             if(routes[i][j] == 999)
503            routes[i][j] = routes[j][i] = 999;
504        }
505    }
506 }
507
508 void run31()
509 {
510     int i, j;
511     for(i = 0; i < size; i++)
512     {
513         for(j =0; j < size; j++)
514         {
515             if(routes[i][j] == 999)
516            routes[i][j] = routes[j][i] = 999;
517        }
518    }
519 }
520
521 void run32()
522 {
523     int i, j;
524     for(i = 0; i < size; i++)
525     {
526         for(j =0; j < size; j++)
527         {
528             if(routes[i][j] == 999)
529            routes[i][j] = routes[j][i] = 999;
530        }
531    }
532 }
533
534 void run35()
535 {
536     int i, j;
537     for(i = 0; i < size; i++)
538     {
539         for(j =0; j < size; j++)
540         {
541             if(routes[i][j] == 999)
542            routes[i][j] = routes[j][i] = 999;
543        }
544    }
545 }
546
547 void run36()
548 {
549     int i, j;
550     for(i = 0; i < size; i++)
551     {
552         for(j =0; j < size; j++)
553         {
554             if(routes[i][j] == 999)
555            routes[i][j] = routes[j][i] = 999;
556        }
557    }
558 }
559
560 void run37()
561 {
562     int i, j;
563     for(i = 0; i < size; i++)
564     {
565         for(j =0; j < size; j++)
566         {
567             if(routes[i][j] == 999)
568            routes[i][j] = routes[j][i] = 999;
569        }
570    }
571 }
572
573 void run38()
574 {
575     int i, j;
576     for(i = 0; i < size; i++)
577     {
578         for(j =0; j < size; j++)
579         {
580             if(routes[i][j] == 999)
581            routes[i][j] = routes[j][i] = 999;
582        }
583    }
584 }
585
586 void run39()
587 {
588     int i, j;
589     for(i = 0; i < size; i++)
590     {
591         for(j =0; j < size; j++)
592         {
593             if(routes[i][j] == 999)
594            routes[i][j] = routes[j][i] = 999;
595        }
596    }
597 }
598
599 void run40()
600 {
601     int i, j;
602     for(i = 0; i < size; i++)
603     {
604         for(j =0; j < size; j++)
605         {
606             if(routes[i][j] == 999)
607            routes[i][j] = routes[j][i] = 999;
608        }
609    }
610 }
611
612 void run41()
613 {
614     int i, j;
615     for(i = 0; i < size; i++)
616     {
617         for(j =0; j < size; j++)
618         {
619             if(routes[i][j] == 999)
620            routes[i][j] = routes[j][i] = 999;
621        }
622    }
623 }
624
625 void run42()
626 {
627     int i, j;
628     for(i = 0; i < size; i++)
629     {
630         for(j =0; j < size; j++)
631         {
632             if(routes[i][j] == 999)
633            routes[i][j] = routes[j][i] = 999;
634        }
635    }
636 }
637
638 void run43()
639 {
640     int i, j;
641     for(i = 0; i < size; i++)
642     {
643         for(j =0; j < size; j++)
644         {
645             if(routes[i][j] == 999)
646            routes[i][j] = routes[j][i] = 999;
647        }
648    }
649 }
650
651 void run44()
652 {
653     int i, j;
654     for(i = 0; i < size; i++)
655     {
656         for(j =0; j < size; j++)
657         {
658             if(routes[i][j] == 999)
659            routes[i][j] = routes[j][i] = 999;
660        }
661    }
662 }
663
664 void run45()
665 {
666     int i, j;
667     for(i = 0; i < size; i++)
668     {
669         for(j =0; j < size; j++)
670         {
671             if(routes[i][j] == 999)
672            routes[i][j] = routes[j][i] = 999;
673        }
674    }
675 }
676
677 void run46()
678 {
679     int i, j;
680     for(i = 0; i < size; i++)
681     {
682         for(j =0; j < size; j++)
683         {
684             if(routes[i][j] == 999)
685            routes[i][j] = routes[j][i] = 999;
686        }
687    }
688 }
689
690 void run47()
691 {
692     int i, j;
693     for(i = 0; i < size; i++)
694     {
695         for(j =0; j < size; j++)
696         {
697             if(routes[i][j] == 999)
698            routes[i][j] = routes[j][i] = 999;
699        }
700    }
701 }
702
703 void run48()
704 {
705     int i, j;
706     for(i = 0; i < size; i++)
707     {
708         for(j =0; j < size; j++)
709         {
710             if(routes[i][j] == 999)
711            routes[i][j] = routes[j][i] = 999;
712        }
713    }
714 }
715
716 void run49()
717 {
718     int i, j;
719     for(i = 0; i < size; i++)
720     {
721         for(j =0; j < size; j++)
722         {
723             if(routes[i][j] == 999)
724            routes[i][j] = routes[j][i] = 999;
725        }
726    }
727 }
728
729 void run50()
730 {
731     int i, j;
732     for(i = 0; i < size; i++)
733     {
734         for(j =0; j < size; j++)
735         {
736             if(routes[i][j] == 999)
737            routes[i][j] = routes[j][i] = 999;
738        }
739    }
740 }
741
742 void run51()
743 {
744     int i, j;
745     for(i = 0; i < size; i++)
746     {
747         for(j =0; j < size; j++)
748         {
749             if(routes[i][j] == 999)
750            routes[i][j] = routes[j][i] = 999;
751        }
752    }
753 }
754
755 void run52()
756 {
757     int i, j;
758     for(i = 0; i < size; i++)
759     {
760         for(j =0; j < size; j++)
761         {
762             if(routes[i][j] == 999)
763            routes[i][j] = routes[j][i] = 999;
764        }
765    }
766 }
767
768 void run53()
769 {
770     int i, j;
771     for(i = 0; i < size; i++)
772     {
773         for(j =0; j < size; j++)
774         {
775             if(routes[i][j] == 999)
776            routes[i][j] = routes[j][i] = 999;
777        }
778    }
779 }
780
781 void run54()
782 {
783     int i, j;
784     for(i = 0; i < size; i++)
785     {
786         for(j =0; j < size; j++)
787         {
788             if(routes[i][j] == 999)
789            routes[i][j] = routes[j][i] = 999;
790        }
791    }
792 }
793
794 void run55()
795 {
796     int i, j;
797     for(i = 0; i < size; i++)
798     {
799         for(j =0; j < size; j++)
800         {
801             if(routes[i][j] == 999)
802            routes[i][j] = routes[j][i] = 999;
803        }
804    }
805 }
806
807 void run56()
808 {
809     int i, j;
810     for(i = 0; i < size; i++)
811     {
812         for(j =0; j < size; j++)
813         {
814             if(routes[i][j] == 999)
815            routes[i][j] = routes[j][i] = 999;
816        }
817    }
818 }
819
820 void run57()
821 {
822     int i, j;
823     for(i = 0; i < size; i++)
824     {
825         for(j =0; j < size; j++)
826         {
827             if(routes[i][j] == 999)
828            routes[i][j] = routes[j][i] = 999;
829        }
830    }
831 }
832
833 void run58()
834 {
835     int i, j;
836     for(i = 0; i < size; i++)
837     {
838         for(j =0; j < size; j++)
839         {
840             if(routes[i][j] == 999)
841            routes[i][j] = routes[j][i] = 999;
842        }
843    }
844 }
845
846 void run59()
847 {
848     int i, j;
849     for(i = 0; i < size; i++)
850     {
851         for(j =0; j < size; j++)
852         {
853             if(routes[i][j] == 999)
854            routes[i][j] = routes[j][i] = 999;
855        }
856    }
857 }
858
859 void run60()
860 {
861     int i, j;
862     for(i = 0; i < size; i++)
863     {
864         for(j =0; j < size; j++)
865         {
866             if(routes[i][j] == 999)
867            routes[i][j] = routes[j][i] = 999;
868        }
869    }
870 }
871
872 void run61()
873 {
874     int i, j;
875     for(i = 0; i < size; i++)
876     {
877         for(j =0; j < size; j++)
878         {
879             if(routes[i][j] == 999)
880            routes[i][j] = routes[j][i] = 999;
881        }
882    }
883 }
884
885 void run62()
886 {
887     int i, j;
888     for(i = 0; i < size; i++)
889     {
890         for(j =0; j < size; j++)
891         {
892             if(routes[i][j] == 999)
893            routes[i][j] = routes[j][i] = 999;
894        }
895    }
896 }
897
898 void run63()
899 {
900     int i, j;
901     for(i = 0; i < size; i++)
902     {
903         for(j =0; j < size; j++)
904         {
905             if(routes[i][j] == 999)
906            routes[i][j] = routes[j][i] = 999;
907        }
908    }
909 }
910
911 void run64()
912 {
913     int i, j;
914     for(i = 0; i < size; i++)
915     {
916         for(j =0; j < size; j++)
917         {
918             if(routes[i][j] == 999)
919            routes[i][j] = routes[j][i] = 999;
920        }
921    }
922 }
923
924 void run65()
925 {
926     int i, j;
927     for(i = 0; i < size; i++)
928     {
929         for(j =0; j < size; j++)
930         {
931             if(routes[i][j] == 999)
932            routes[i][j] = routes[j][i] = 999;
933        }
934    }
935 }
936
937 void run66()
938 {
939     int i, j;
940     for(i = 0; i < size; i++)
941     {
942         for(j =0; j < size; j++)
943         {
944             if(routes[i][j] == 999)
945            routes[i][j] = routes[j][i] = 999;
946        }
947    }
948 }
949
950 void run67()
951 {
952     int i, j;
953     for(i = 0; i < size; i++)
954     {
955         for(j =0; j < size; j++)
956         {
957             if(routes[i][j] == 999)
958            routes[i][j] = routes[j][i] = 999;
959        }
960    }
961 }
962
963 void run68()
964 {
965     int i, j;
966     for(i = 0; i < size; i++)
967     {
968         for(j =0; j < size; j++)
969         {
970             if(routes[i][j] == 999)
971            routes[i][j] = routes[j][i] = 999;
972        }
973    }
974 }
975
976 void run69()
977 {
978     int i, j;
979     for(i = 0; i < size; i++)
980     {
981         for(j =0; j < size; j++)
982         {
983             if(routes[i][j] == 999)
984            routes[i][j] = routes[j][i] = 999;
985        }
986    }
987 }
988
989 void run70()
990 {
991     int i, j;
992     for(i = 0; i < size; i++)
993     {
994         for(j =0; j < size; j++)
995         {
996             if(routes[i][j] == 999)
997            routes[i][j] = routes[j][i] = 999;
998        }
999    }
1000 }
1001
1002 void run71()
1003 {
1004     int i, j;
1005     for(i = 0; i < size; i++)
1006     {
1007         for(j =0; j < size; j++)
1008         {
1009             if(routes[i][j] == 999)
1010            routes[i][j] = routes[j][i] = 999;
1011        }
1012    }
1013 }
1014
1015 void run72()
1016 {
1017     int i, j;
1018     for(i = 0; i < size; i++)
1019     {
1020         for(j =0; j < size; j++)
1021         {
1022             if(routes[i][j] == 999)
1023            routes[i][j] = routes[j][i] = 999;
1024        }
1025    }
1026 }
1027
1028 void run73()
1029 {
1030     int i, j;
1031     for(i = 0; i < size; i++)
1032     {
1033         for(j =0; j < size; j++)
1034         {
1035             if(routes[i][j] == 999)
1036            routes[i][j] = routes[j][i] = 999;
1037        }
1038    }
1039 }
1040
1041 void run74()
1042 {
1043     int i, j;
1044     for(i = 0; i < size; i++)
1045     {
1046         for(j =0; j < size; j++)
1047         {
1048             if(routes[i][j] == 999)
1049            routes[i][j] = routes[j][i] = 999;
1050        }
1051    }
1052 }
1053
1054 void run75()
1055 {
1056     int i, j;
1057     for(i = 0; i < size; i++)
1058     {
1059         for(j =0; j < size; j++)
1060         {
1061             if(routes[i][j] == 999)
1062            routes[i][j] = routes[j][i] = 999;
1063        }
1064    }
1065 }
1066
1067 void run76()
1068 {
1069     int i, j;
1070     for(i = 0; i < size; i++)
1071     {
1072         for(j =0; j < size; j++)
1073         {
1074             if(routes[i][j] == 999)
1075            routes[i][j] = routes[j][i] = 999;
1076        }
1077    }
1078 }
1079
1080 void run77()
1081 {
1082     int i, j;
1083     for(i = 0; i < size; i++)
1084     {
1085         for(j =0; j < size; j++)
1086         {
1087             if(routes[i][j] == 999)
1088            routes[i][j] = routes[j][i] = 999;
1089        }
1090    }
1091 }
1092
1093 void run78()
1094 {
1095     int i, j;
1096     for(i = 0; i < size; i++)
1097     {
1098         for(j =0; j < size; j++)
1099         {
1100             if(routes[i][j] == 999)
1101            routes[i][j] = routes[j][i] = 999;
1102        }
1103    }
1104 }
1105
1106 void run79()
1107 {
1108     int i, j;
1109     for(i = 0; i < size; i++)
1110     {
1111         for(j =0; j < size; j++)
1112         {
1113             if(routes[i][j] == 999)
1114            routes[i][j] = routes[j][i] = 999;
1115        }
1116    }
1117 }
1118
1119 void run80()
1120 {
1121     int i, j;
1122     for(i = 0; i < size; i++)
1123     {
1124         for(j =0; j < size; j++)
1125         {
1126             if(routes[i][j] == 999)
1127            routes[i][j] = routes[j][i] = 999;
1128        }
1129    }
1130 }
1131
1132 void run81()
1133 {
1134     int i, j;
1135     for(i = 0; i < size; i++)
1136     {
1137         for(j =0; j < size; j++)
1138         {
1139             if(routes[i][j] == 999)
1140            routes[i][j] = routes[j][i] = 999;
1141        }
1142    }
1143 }
1144
1145 void run82()
1146 {
1147     int i, j;
1148     for(i = 0; i < size; i++)
1149     {
1150         for(j =0; j < size; j++)
1151         {
1152             if(routes[i][j] == 999)
1153            routes[i][j] = routes[j][i] = 999;
1154        }
1155    }
1156 }
1157
1158 void run83()
1159 {
1160     int i, j;
1161     for(i = 0; i < size; i++)
1162     {
1163         for(j =0; j < size; j++)
1164         {
1165             if(routes[i][j] == 999)
1166            routes[i][j] = routes[j][i] = 999;
1167        }
1168    }
1169 }
1170
1171 void run84()
1172 {
1173     int i, j;
1174     for(i = 0; i < size; i++)
1175     {
1176         for(j =0; j < size; j++)
1177         {
1178             if(routes[i][j] == 999)
1179            routes[i][j] = routes[j][i] = 999;
1180        }
1181    }
1182 }
1183
1184 void run85()
1185 {
1186     int i, j;
1187     for(i = 0; i < size; i++)
1188     {
1189         for(j =0; j < size; j++)
1190         {
1191             if(routes[i][j] == 999)
1192            routes[i][j] = routes[j][i] = 999;
1193        }
1194    }
1195 }
1196
1197 void run86()
1198 {
1199     int i, j;
1200     for(i = 0; i < size; i++)
1201     {
1202         for(j =0; j < size; j++)
1203         {
1204             if(routes[i][j] == 999)
1205            routes[i][j] = routes[j][i] = 999;
1206        }
1207    }
1208 }
1209
1210 void run87()
1211 {
1212     int i, j;
1213     for(i = 0; i < size; i++)
1214     {
1215         for(j =0; j < size; j++)
1216         {
1217             if(routes[i][j] == 999)
1218            routes[i][j] = routes[j][i] = 999;
1219        }
1220    }
1221 }
1222
1223 void run88()
1224 {
1225     int i, j;
1226     for(i = 0; i
```

Updates Available

Updates are available for your software.
Click to review and install updates.

Set up [Reminder options](#)

```

    BTREE-2046539460.minibase-db
    BTREE-208838911.minibase-db
    BTREE-209772834.minibase-db
    BTREE-2193454438.minibase-db
    BTREE-227853136.minibase-db
    BTREE-347026083.minibase-db
    BTREE-401690725.minibase-db
    BTREE-448278888.minibase-db
    BTREE-497824974.minibase-db
    BTREE-666962870.minibase-db
    BTREE-735569689.minibase-db
    BTREE-905769395.minibase-db
    BTREE1004400680.minibase-db
    BTREE1073189410.minibase-db
    BTREE1159929293.minibase-db
    BTREE1592187132.minibase-db
    BTREE1829384116.minibase-db
    BTREE1892500505.minibase-db
    BTREE1977580190.minibase-db
    BTREE2043574380.minibase-db
    BTREE231397800.minibase-db
    BTREE234426026.minibase-db
    BTREE302972527.minibase-db
    BTREE427285043.minibase-db
    BTREE464030461.minibase-db
    BTREE479048644.minibase-db
    BTREEB04384892.minibase-db
    BTREEB37205469.minibase-db
    BTREEB66040111.minibase-db
    BTREE918293212.minibase-db
  
```

TravellingSalesman

- > IRE System Library [JavaSE-1.8]
- > src

traverses

- > IRE System Library [JavaSE-1.8]
- > src

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 14:25)

Writable Smart Insert 226:1 405 PM 2/7/2019

Updates Available

Updates are available for your software.
Click to review and install updates.

Set up [Reminder options](#)

```

    BTREE-2046539460.minibase-db
    BTREE-208838911.minibase-db
    BTREE-209772834.minibase-db
    BTREE-2193454438.minibase-db
    BTREE-227853136.minibase-db
    BTREE-347026083.minibase-db
    BTREE-401690725.minibase-db
    BTREE-448278888.minibase-db
    BTREE-497824974.minibase-db
    BTREE-666962870.minibase-db
    BTREE-735569689.minibase-db
    BTREE-905769395.minibase-db
    BTREE1004400680.minibase-db
    BTREE1073189410.minibase-db
    BTREE1159929293.minibase-db
    BTREE1592187132.minibase-db
    BTREE1829384116.minibase-db
    BTREE1892500505.minibase-db
    BTREE1977580190.minibase-db
    BTREE2043574380.minibase-db
    BTREE231397800.minibase-db
    BTREE234426026.minibase-db
    BTREE302972527.minibase-db
    BTREE427285043.minibase-db
    BTREE464030461.minibase-db
    BTREE479048644.minibase-db
    BTREEB04384892.minibase-db
    BTREEB37205469.minibase-db
    BTREEB66040111.minibase-db
    BTREE918293212.minibase-db
  
```

TravellingSalesman

- > IRE System Library [JavaSE-1.8]
- > src

traverses

- > IRE System Library [JavaSE-1.8]
- > src

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 14:25)

Writable Smart Insert 226:1 405 PM 2/7/2019

Updates Available

Updates are available for your software.
Click to review and install updates.

Set up [Reminder options](#)

```

145         }
146     } //for(i ...
147
148     if(flag == 0)
149     { //unvisited
150         if(routes[currentPos][i] < min)
151         {
152             min = routes[currentPos][i];
153             nextPos = i;
154         }
155     }
156 } //if(routes...
157
158 } //for(i ...
159
160 if(nextPos != -1)
161 { //move to next city
162     totDistance += min;
163     visitedCities[v1] = nextPos;
164     v1++;
165     solution = solution + " " + cities[nextPos];
166     currentPos = nextPos;
167 }
168 else
169 {
170     break;
171 }
172
173 if(vi == size)
174 {
175     //tour back to start city
176     if(routes[currentPos][startPos] != 999)
177     {
178         solution = solution + " " + cities[startPos];
179         totDistance += routes[currentPos][startPos];
180         isTourComplete = true;
181         if(totDistance < this.totDistance)
182         {
183             this.totDistance = totDistance;
184             this.solution = solution + "\nTotal Distance : " + totDistance;
185         }
186     }
187     else
188     {
189         isTourComplete = false;
190         break;
191     }
192 } //while
193
194 }
195 catch(Exception ex)
196 {
197     solution = "Err : " + ex.getMessage();
198 }
199
200 return isTourComplete;
201 }
202
203 void displaySolution()
204 {
205 }
```

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 14:25)

Writable Smart Insert 226:1 405 PM 2/7/2019

Updates Available

Updates are available for your software.
Click to review and install updates.

Set up [Reminder options](#)

```

175         }
176     } //for(i ...
177
178     if(routes[currentPos][startPos] != 999)
179     {
180         solution = solution + " " + cities[startPos];
181         totDistance += routes[currentPos][startPos];
182         isTourComplete = true;
183         if(totDistance < this.totDistance)
184         {
185             this.totDistance = totDistance;
186             this.solution = solution + "\nTotal Distance : " + totDistance;
187         }
188     }
189     else
190     {
191         isTourComplete = false;
192         break;
193     }
194 } //while
195
196 catch(Exception ex)
197 {
198     solution = "Err : " + ex.getMessage();
199 }
200
201 return isTourComplete;
202
203 void displaySolution()
204 {
205 }
```

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 14:25)

Writable Smart Insert 226:1 405 PM 2/7/2019

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** Updates Available, Search, Project, Run, Window, Help.
- Code Editor:** File /src/tsp/tsp.java - Eclipse IDE. The code is for a Travelling Salesman problem, specifically a class named `tsp`.
- SonarLint Results:**
 - 37 items reported in total.
 - 21 major code smells.
 - 15 minor code smells.
 - 1 critical code smell.
- Outline View:** Shows the class structure with methods `isOurComplete`, `displaySolution`, and `main`.
- Problems View:** Lists specific code smell violations with descriptions and suggested fixes.
- Bottom Status Bar:** Writable, Smart Insert, 226:1, 405 PM, 2/7/2019.

Tool used: SonarLint 4.1

Severity of SonarLint markers: Info

Code Smell:

1) Number of major Code Smells: 21

2) Number of minor Code Smells: 15

3) Number of critical Code Smells: 1

37 items reported in total

Code Smell Fixes done:

Lines 91,58,59,156,158, 146:

Issue was: This block of commented-out lines of code should be removed.

Lines 108,82,21:

Issue was: Variable Declaration should be done on a separate line.

Screenshots of fixes done:

eclipse-workspace - TravellingSalesman/src/tsp/tsp.java - Eclipse IDE

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Package Explorer

tsp.java

```
1 package tsp;
2
3 import java.util.*;
4
5 public class tsp implements Runnable
6 {
7     int size;
8     int routes[][];//edges
9     String cities[];//vertices
10    Thread runners[];
11    int threadCompleteCount;
12
13    String solution;
14    int totDistance;
15
16    tsp()
17    {
18        try
19        {
20            int i;
21            int j;
22            String choice;
23
24            Scanner scan = new Scanner(System.in);
25            System.out.println("Enter the number of cities ");
26
27            size = scan.nextInt();
28            scan.skip("\n");
29
30            cities = new String[size];
31            runners = new Thread[threadCompleteCount];
32
33            for(i=0; i<size; i++)
34            {
35                System.out.print("City " + (i+1) );
36                cities[i] = scan.nextLine();
37            }
38
39            System.out.println("Set interconnecting routes ");
40            for(i=0; i<size; i++)
41            {
42
43                for(j=i+1; j<size; j++)
44                {
45                    System.out.println("Is there a route between " + cities[i] + " and " + cities[j] + "(y/n) ");
46                    choice = scan.nextLine();
47                    if(choice.equalsIgnoreCase("y"))
48                    {
49                        System.out.print("Enter distance : ");
50                        routes[i][j] = routes[j][i] = scan.nextInt();
51                        scan.skip("\n");
52                    }
53                    else
54                    {
55                        routes[i][j] = routes[j][i] = 999;
56                    }
57                }
58            }
59        }
60    }
61}
```

Problems Javadoc Declaration Search Coverage SonarLint Report Bug Explorer Bug Info SonarLint Rule Description

25 items

Resource	Date	Description
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Reduce the total number of break and continue statements in this loop to use at most one.

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)

Writable Smart Insert 114:25 457 PM 2/7/2019

eclipse-workspace - TravellingSalesman/src/tsp/tsp.java - Eclipse IDE

```
File Edit Source Refactor Navigate Search Project Run Window Help
```

Package Explorer

tsp.java

```
31     cities = new String[size];
32     routes = new int[size][size];
33
34     System.out.println("Set city names : ");
35     for(i=0; i<size; i++)
36     {
37         System.out.print("City " + (i+1) );
38         cities[i] = scan.nextLine();
39     }
40
41     System.out.println("Set interconnecting routes ");
42     for(i=0; i<size; i++)
43     {
44
45         for(j=i+1; j<size; j++)
46         {
47             System.out.println("Is there a route between " + cities[i] + " and " + cities[j] + "(y/n) ");
48             choice = scan.nextLine();
49             if(choice.equalsIgnoreCase("y"))
50             {
51                 System.out.print("Enter distance : ");
52                 routes[i][j] = routes[j][i] = scan.nextInt();
53                 scan.skip("\n");
54             }
55             else
56             {
57                 routes[i][j] = routes[j][i] = 999;
58             }
59         }
60     }
61}
```

Problems Javadoc Declaration Search Coverage SonarLint Report Bug Explorer Bug Info SonarLint Rule Description

25 items

Resource	Date	Description
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Reduce the total number of break and continue statements in this loop to use at most one.

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)

Writable Smart Insert 114:25 457 PM 2/7/2019

eclipse-workspace - TravellingSalesman/src/tsp/tsp.java - Eclipse IDE

```

File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer  tsjava
BTREEE-2046539460.minibase-db
BTREEE-20838911.minibase-db
BTREEE-209772834.minibase-db
BTREEE-219454438.minibase-db
BTREEE-227855136.minibase-db
BTREEE-347026083.minibase-db
BTREEE-401690725.minibase-db
BTREEE-448278888.minibase-db
BTREEE-462320919.minibase-db
BTREEE-497824974.minibase-db
BTREEE-666962870.minibase-db
BTREEE-735569689.minibase-db
BTREEE-905769395.minibase-db
BTREEE1004408660.minibase-db
BTREEE1073189410.minibase-db
BTREEE1159929293.minibase-db
BTREEE152187132.minibase-db
BTREEE1774386416.minibase-db
BTREEE1892500505.minibase-db
BTREEE1977580190.minibase-db
BTREEE2043574380.minibase-db
BTREEE231397800.minibase-db
BTREEE2344262026.minibase-db
BTREEE302972527.minibase-db
BTREEE427285043.minibase-db
BTREEE46403461.minibase-db
BTREEE479048644.minibase-db
BTREEB004384892.minibase-db
BTREEB37205469.minibase-db
BTREEB66640111.minibase-db
BTREE918293212.minibase-db
TravellingSalesman
  IRE System Library [JavaSE-1.8]
    src
traverses
  IRE System Library [JavaSE-1.8]
    src
File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)

```

Problems Javadoc Declaration Search Coverage SonarLint Report Bug Explorer Bug Info SonarLint Rule Description

25 items

Resource	Date	Description
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Reduce the total number of break and continue statements in this loop to use at most one.

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)

eclipse-workspace - TravellingSalesman/src/tsp/tsp.java - Eclipse IDE

```

File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer  tsjava
BTREEE-2046539460.minibase-db
BTREEE-20838911.minibase-db
BTREEE-209772834.minibase-db
BTREEE-219454438.minibase-db
BTREEE-227855136.minibase-db
BTREEE-347026083.minibase-db
BTREEE-401690725.minibase-db
BTREEE-448278888.minibase-db
BTREEE-462320919.minibase-db
BTREEE-497824974.minibase-db
BTREEE-666962870.minibase-db
BTREEE-735569689.minibase-db
BTREEE-905769395.minibase-db
BTREEE1004408660.minibase-db
BTREEE1073189410.minibase-db
BTREEE1159929293.minibase-db
BTREEE152187132.minibase-db
BTREEE1774386416.minibase-db
BTREEE1892500505.minibase-db
BTREEE1977580190.minibase-db
BTREEE2043574380.minibase-db
BTREEE231397800.minibase-db
BTREEE2344262026.minibase-db
BTREEE302972527.minibase-db
BTREEE427285043.minibase-db
BTREEE46403461.minibase-db
BTREEE479048644.minibase-db
BTREEB004384892.minibase-db
BTREEB37205469.minibase-db
BTREEB66640111.minibase-db
BTREE918293212.minibase-db
TravellingSalesman
  IRE System Library [JavaSE-1.8]
    src
traverses
  IRE System Library [JavaSE-1.8]
    src
File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)

```

Problems Javadoc Declaration Search Coverage SonarLint Report Bug Explorer Bug Info SonarLint Rule Description

25 items

Resource	Date	Description
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Move the array designator from the variable to the type.
tsp.java	2 hours ago	Reduce the total number of break and continue statements in this loop to use at most one.

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)

eclipse-workspace - TravellingSalesman/src/tsp/tsp.java - Eclipse IDE

```
//initializations and allocations
visitedCities = new int[size];
vi = 0;
totDistance = 0;

//mark startPos as visited
visitedCities[vi] = startPos;
vi++;

solution = cities[startPos];

currentPos = startPos;
//tour

while(!isTourComplete)
{
    nextPos = -1;
    min = 999;

    for(i=0; i < size; i++)
    {
        if(routes[currentPos][i] != 999 && currentPos != i)
        {
            int flag = 0;

            //check for being unvisited
            for(j=0; j < vi; j++)
            {
                if(visitedCities[j] == i)
                    ...
            }
        }
    }
}

if(nextPos != -1)
{
    flag = 1;
    break;
}

if(flag == 0)
//unvisited
if(routes[currentPos][i] < min)
{
    min = routes[currentPos][i];
    nextPos = i;
}
}

if(nextPos != -1)
//move to next city
totDistance += min;
visitedCities[vi] = nextPos;
vi++;
solution = solution + " " + cities[nextPos];
currentPos = nextPos;
}
else
{
    break;
}
```

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)

eclipse-workspace - TravellingSalesman/src/tsp/tsp.java - Eclipse IDE

```
//initializations and allocations
visitedCities = new int[size];
vi = 0;
totDistance = 0;

//mark startPos as visited
visitedCities[vi] = startPos;
vi++;

solution = cities[startPos];

currentPos = startPos;
//tour

while(!isTourComplete)
{
    nextPos = -1;
    min = 999;

    for(i=0; i < size; i++)
    {
        if(routes[currentPos][i] != 999 && currentPos != i)
        {
            int flag = 0;

            //check for being unvisited
            for(j=0; j < vi; j++)
            {
                if(visitedCities[j] == i)
                    ...
            }
        }
    }
}

if(nextPos != -1)
{
    flag = 1;
    break;
}

if(flag == 0)
//unvisited
if(routes[currentPos][i] < min)
{
    min = routes[currentPos][i];
    nextPos = i;
}
}

if(nextPos != -1)
//move to next city
totDistance += min;
visitedCities[vi] = nextPos;
vi++;
solution = solution + " " + cities[nextPos];
currentPos = nextPos;
}
else
{
    break;
}
```

File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)

The screenshot shows the Eclipse IDE interface with the following details:

- Top Bar:** eclipse-workspace - TravellingSalesman/src/tsp/tsp.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Project Run Window Help
- Toolbar:** Standard Java development toolbar with icons for file operations, search, and navigation.
- Left Sidebar:** Package Explorer and TravellingSalesman project structure.
- Central Area:** Code editor for `tsp.java` containing the provided Java code for the Traveling Salesman Problem.
- Right Sidebar:** Quick Access, Task List, Find/Replace, Outline, Problems, Javadoc, Declaration, Coverage, SonarLint Report, Bug Explorer, Bug Info, and SonarLint Rule Description.

```
177         break;
178     }
179     }
180     }
181     if(vi == size)
182     {
183         //tour back to start city
184         if(routes[currentPos][startPos] != 999)
185         {
186             solution = solution + " " + cities[startPos];
187             totDistance += routes[currentPos][startPos];
188             isTourComplete = true;
189             if(totDistance < this.totDistance)
190             {
191                 this.totDistance = totDistance;
192                 this.solution = solution + "\nTotal Distance : " + totDistance;
193             }
194         }
195         else
196         {
197             isTourComplete = false;
198             break;
199         }
200     } //while
201 }
202 }
203 catch(Exception ex)
204 {
205     solution = "Err : " + ex.getMessage();
206 }
207 return isTourComplete;
208 }
```

A screenshot of the Eclipse IDE interface. The title bar shows "eclipse-workspace - TravellingSalesman/src/tsp/tsp.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Project, Run, Window, Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, etc. The left sidebar has a Package Explorer with a tree view of files under "TravellingSalesman". The main editor window displays the code for "tsp.java". The code implements a solution for a Traveling Salesman Problem using a minibase database and a thread-based approach. It includes methods for catching exceptions, displaying solutions, and a main method that creates a TSP object, displays it, and then displays its solution. The right sidebar contains a Quick Access bar, a Task List, and an Outline view showing the class structure. The bottom status bar shows "File /TravellingSalesman/src/tsp/tsp.java (at 07/02/2019 16:56)" and "Writable".

Thus, as seen from the screenshots the number of reported items dropped from 37 to 25 due to the fixes mentioned above.